CSE 559A: Computer Vision

[credit: danjodon.deviantart.com]

Fall 2017: T-R: 11:30-1pm @ Lopata 101

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Staff: Abby Stylianou (abby@wustl.edu), Jarett Gross (jarett@wustl.edu)

http://www.cse.wustl.edu/~ayan/courses/cse559a/

Oct 12, 2017

GENERAL

- Problem set 3 posted Tuesday. Due two weeks from Today.

- No Class Tuesday (Fall Break)

- No office hours tomorrow or monday. Recitation next Friday.
    - May have additional office hours before problem set is due.

- Collect PSET1 Key in class.

FURTHER READING (OPTIONAL)

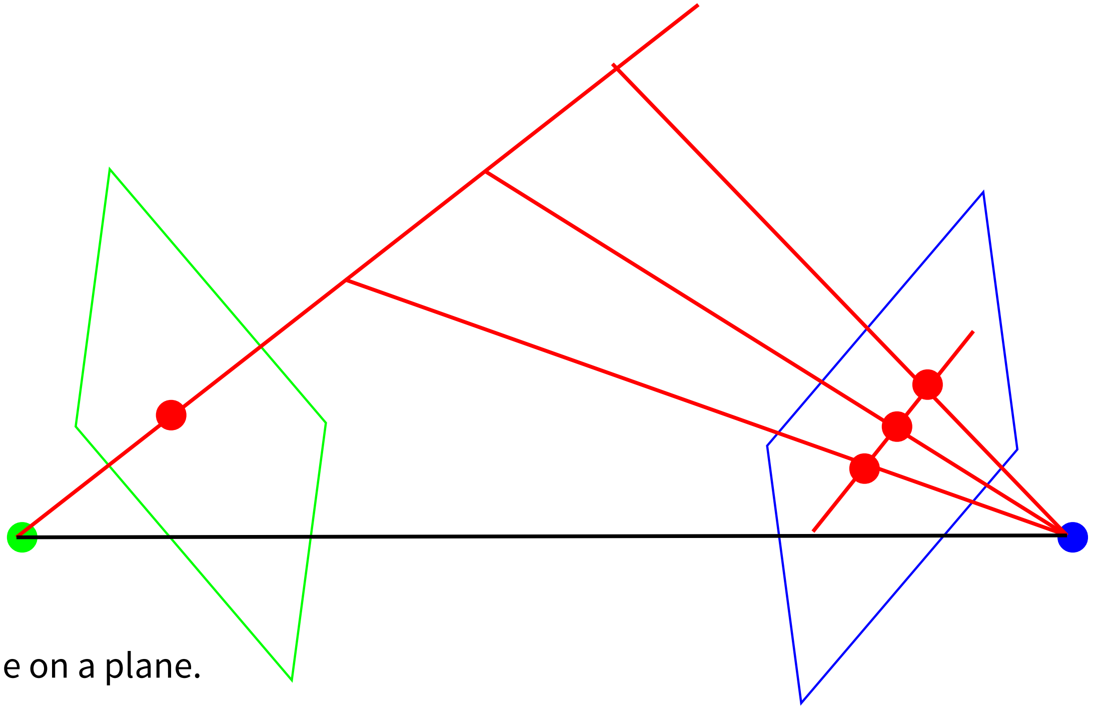
Szeliski Section 2.1

Szeliski Chapter 6

- Other means of camera calibration
- Minimizing other error metrics
- Lens Distortions and Dealing with them

TWO VIEW GEOMETRY

Point in the world can lie anywhere on this line



All points in this picture lie on a plane.

The point in the world must lie on the same plane
as the two camera centers, and the point on the image plane.

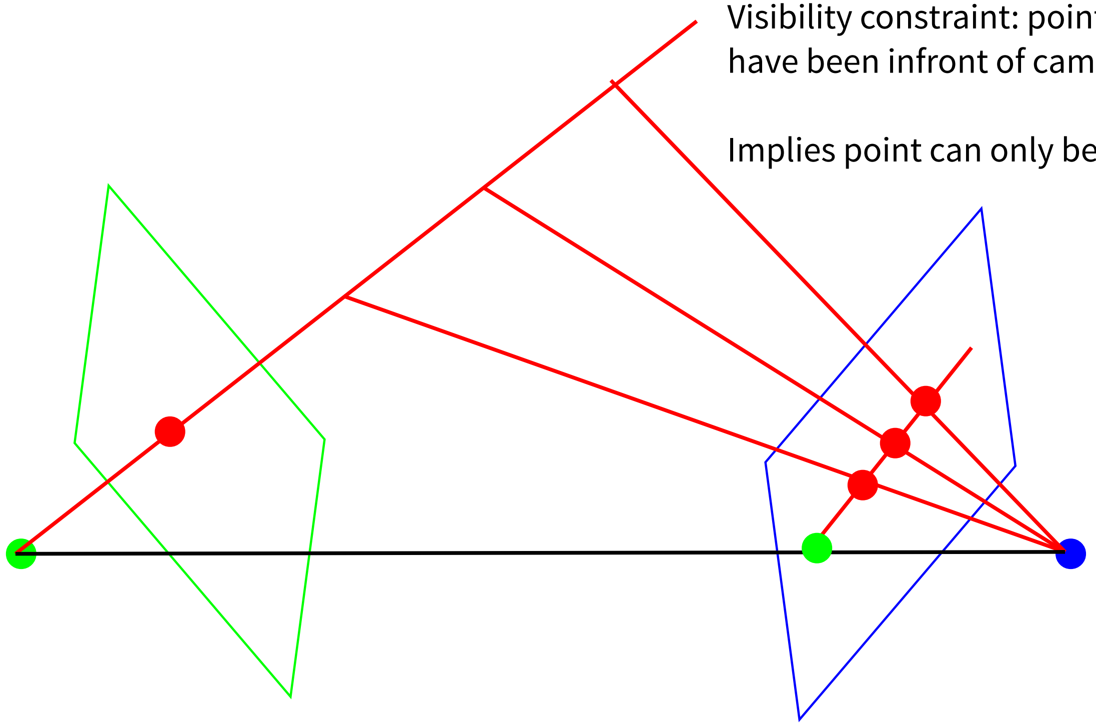
The epipolar line is the intersection of this plane with the second
camera's sensor plane.

All possible points lie on
a line. This is called the
epipolar line corresponding
to the point.

TWO VIEW GEOMETRY

Visibility constraint: point could only
have been infront of camera center.

Implies point can only be on one side of epipole.

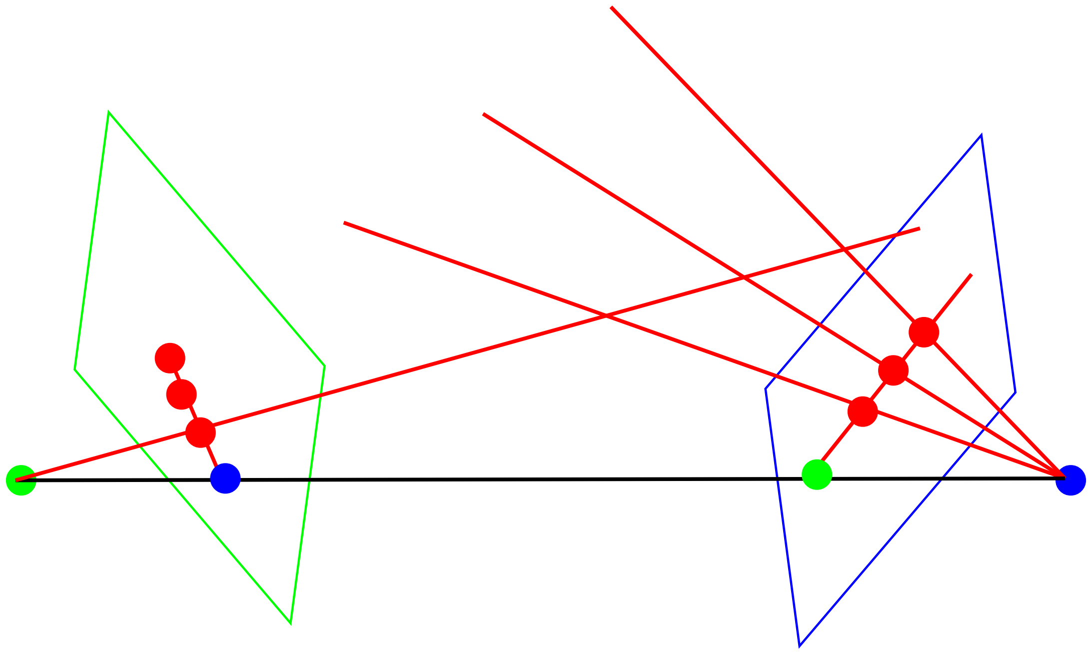


The image of the first camera's center
in the second camera, will lie on this line.

It will lie on all lines. Called the epipole.

All possible points lie on
a line. This is called the
epipolar line corresponding
to the point.

TWO VIEW GEOMETRY



For any of those points on the right image,
there is a corresponding epipolar line in the left

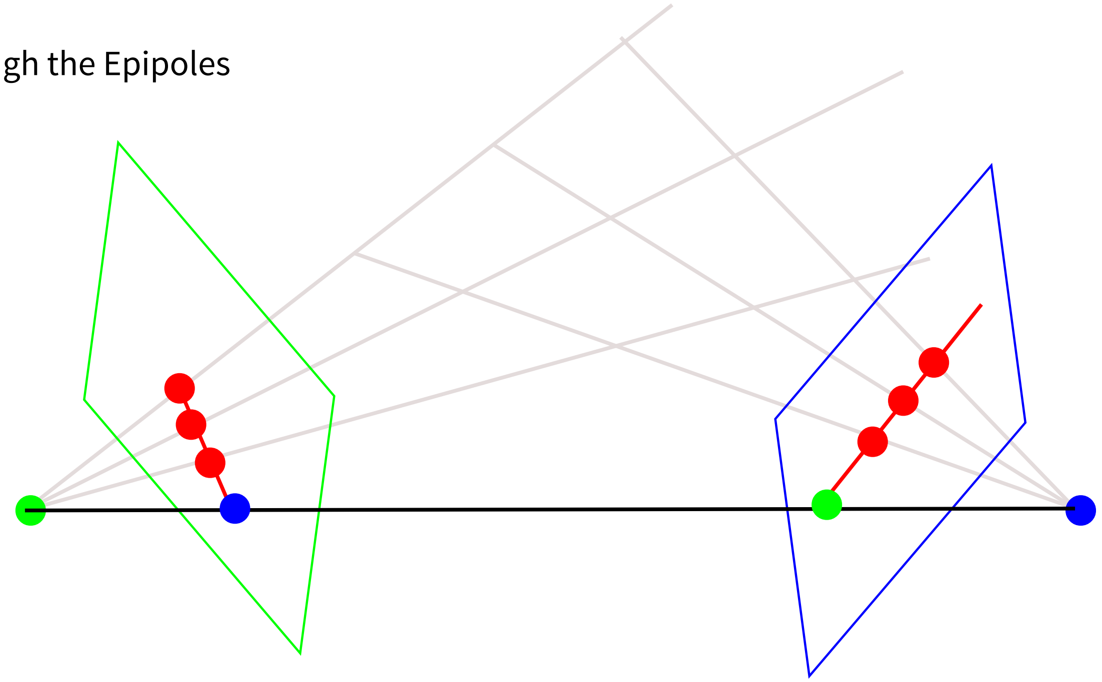
And all points on that line will match to the same epipolar line on the right

All possible points lie on a line. This is called the epipolar line corresponding to the point.

TWO VIEW GEOMETRY

Epipolar Geometry: Lines Match to Lines

All Epipolar Lines go through the Epipoles



For any of those points on the right image,
there is a corresponding epipolar line in the left

And all points on that line will match to the same epipolar line on the right

All possible points lie on a line. This is called the epipolar line corresponding to the point.

TWO VIEW GEOMETRY

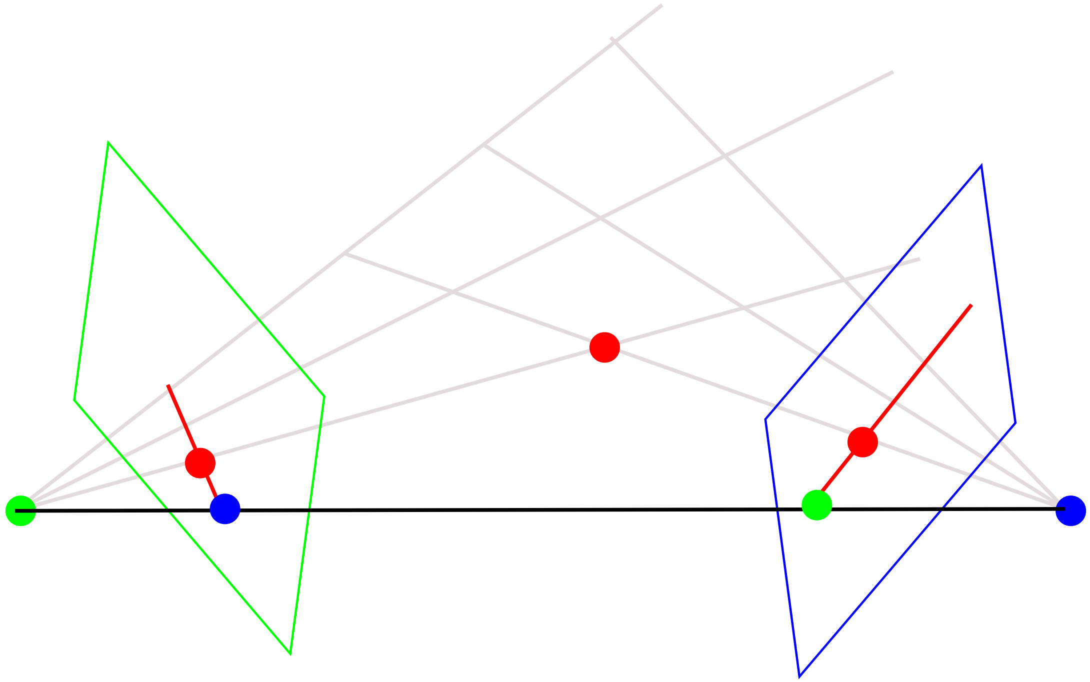
$$p_l^T F p_r = 0$$

- p_l, p_r are 2D homogenous co-ordinates of left and right points.
- Co-ordinates in "image space". F is called the **fundamental matrix**.
- So given a specific point p_r, says $p_l^T(Fp_r) = 0$.
  - This is the equation of a line !
- Same for the other way round.
- Has rank 2. Why ?
- Vector p such that $Fp = [0, 0, 0]^T$.
- Means that this vector p will satisfy $p_l^T Fp$ for every p_l.
- p is the homogeneous co-ordinate for the epipole in the right image.

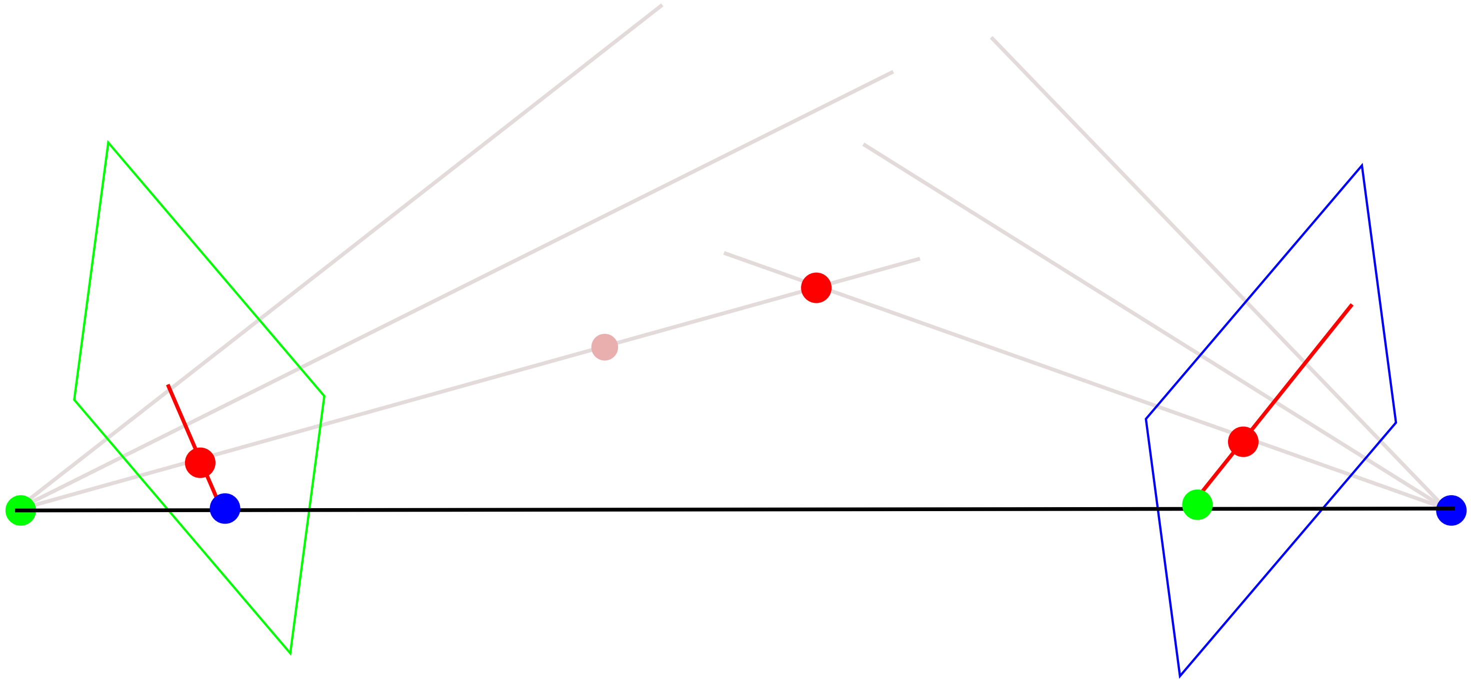
TWO VIEW GEOMETRY

- Fundamental matrix has seven free parameters.
- One free parameter from scale.
- Require that $\det(F) = 0$
- Estimate using correspondences.
    - (see "eight point algorithm" in Szeliski 7.2 / Wikipedia)
- If both cameras are calibrated, then only five unknowns
- Three for rotation
- Only two for translation !
- Only direction of translation matters. Epipolar lines stay the same irrespective of magnitude.

TWO VIEW GEOMETRY



TWO VIEW GEOMETRY

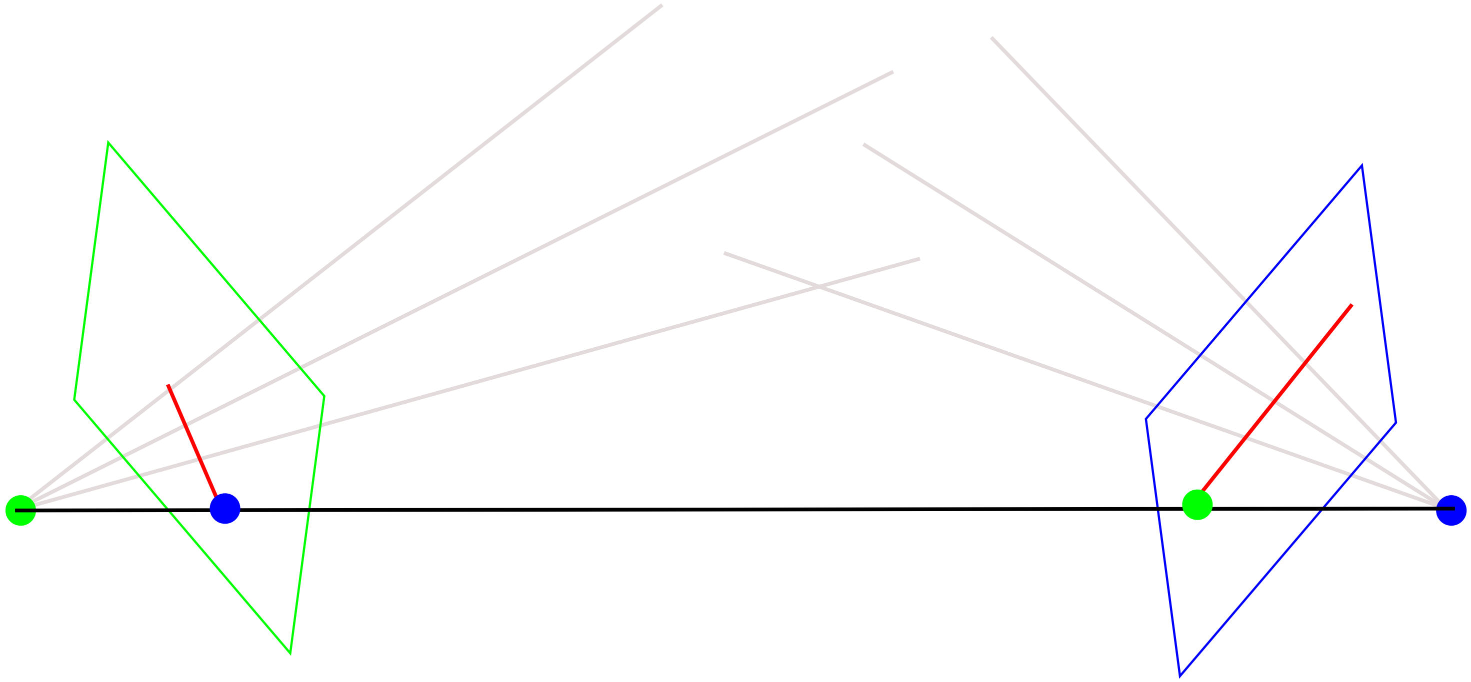


Epipolar Lines stay the same: relationship between depth and location changes.

If you compute fundamental matrix from image correspondences, you can use matching points to get depth only upto this unknown scale corresponding to translation magnitude.

TWO VIEW GEOMETRY

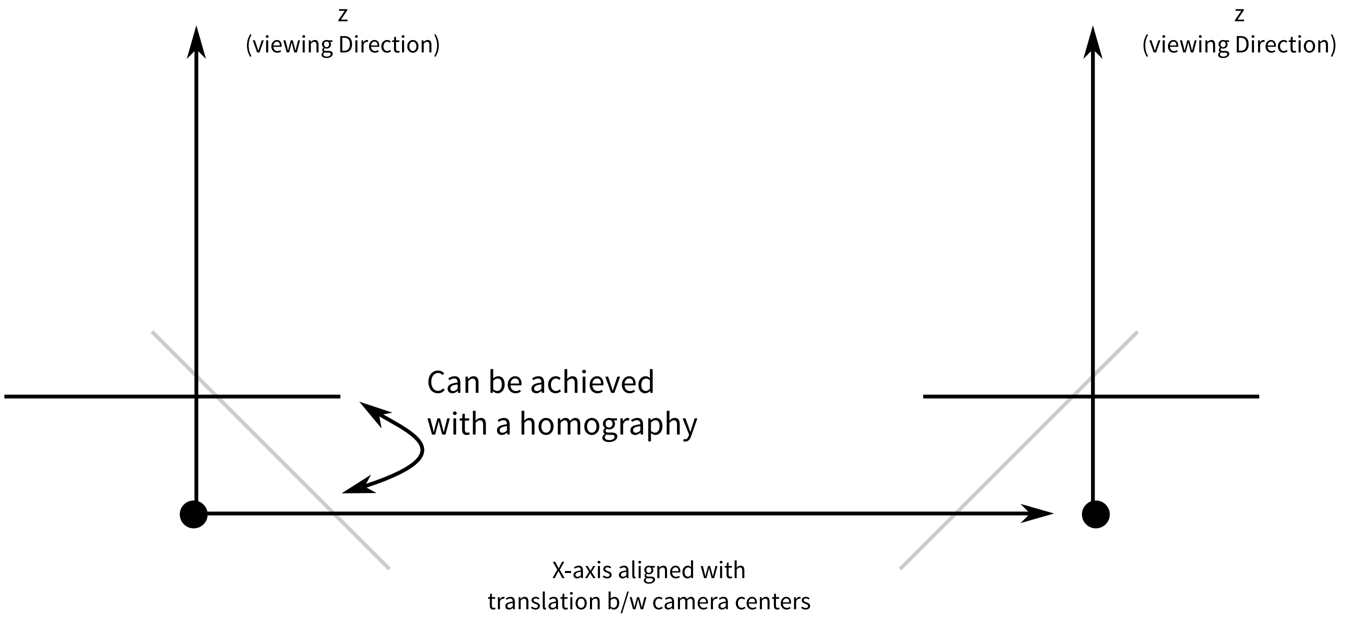
Rectification



Having arbitrary lines is a little annoying.

TWO VIEW GEOMETRY

Rectification

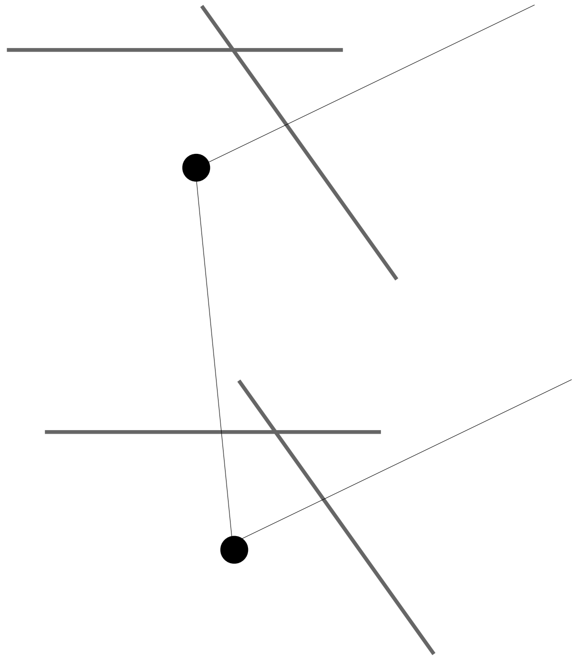


Epipoles at infinity
Epipolar lines all parallel to the X axis

If the cameras were related only by translation,
and viewing direction was orthogonal to the translation vector.

TWO VIEW GEOMETRY

Rectification



Almost always, not ideal
if camera moved in viewing direction
Too much distortion / non-overlap.

the X axis

and viewing direction was orthogonal to the translation vector.

STEREO

Standard Stereo Setting: Rectified Images (or from cameras with pure horizontal translation)

(x,y) in the left image will map to some (x-d,y) in the right image, for some non-negative d.

Left Image

d depends on depth.

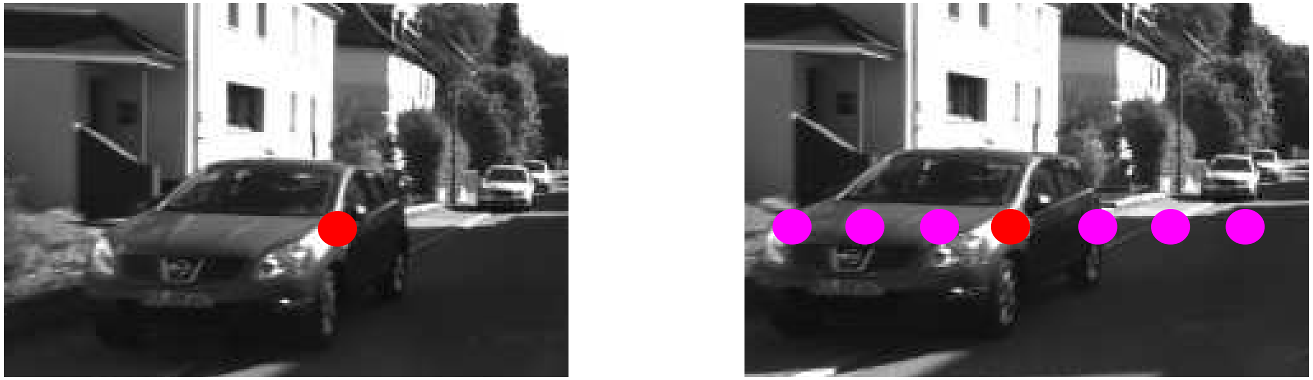
0 for a point at infinity.

Right Image

Challenge in stereo is to find d at each point.

Matching for different values of d (from 0 to some max displacement)

STEREO

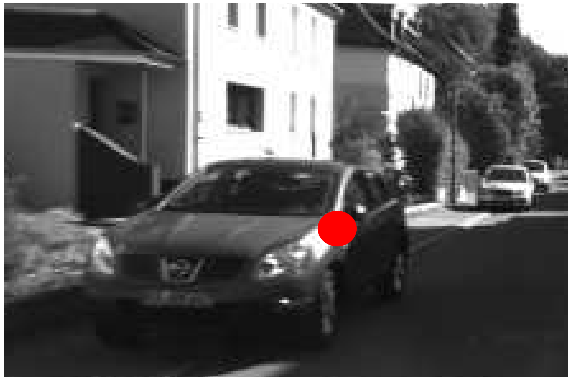


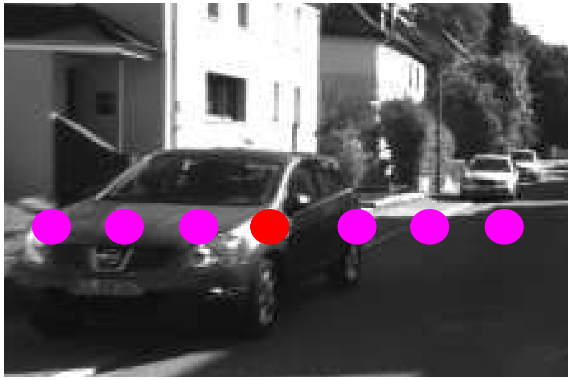
Step 1: At each location in left and right image, compute some representation of the appearance of (neighborhood around) that location.

Step 2: Define a "distance" function.

Step 3: Determine disparity as the "best match" according to this distance.

STEREO

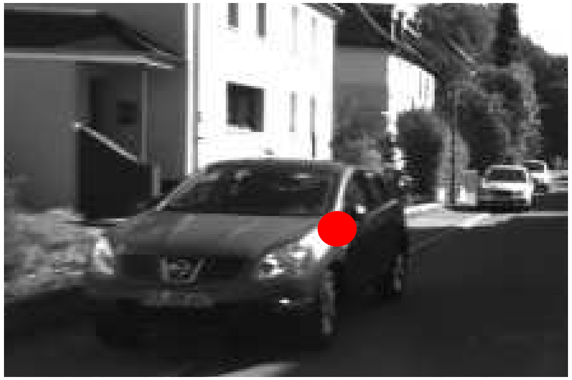


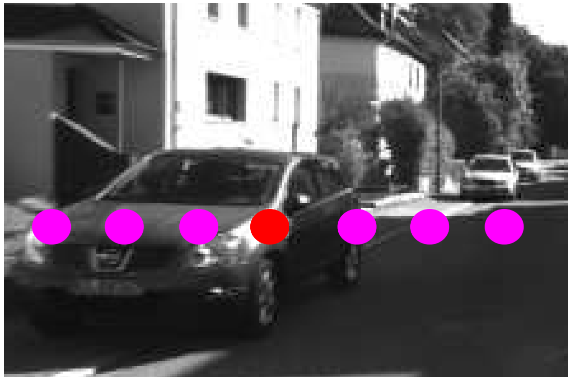


Option 1: Encoding = Intensity, Distance = Absolute Value

Noisy, unstable, susceptible to specular highlights

STEREO

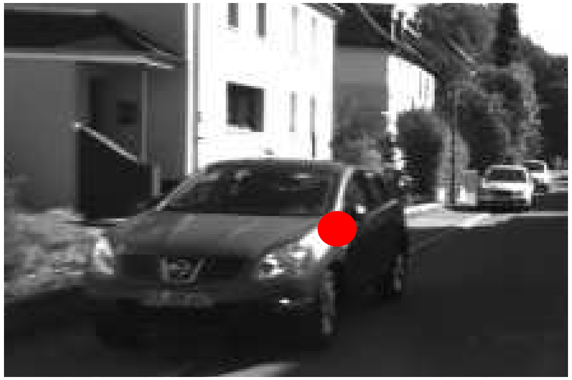


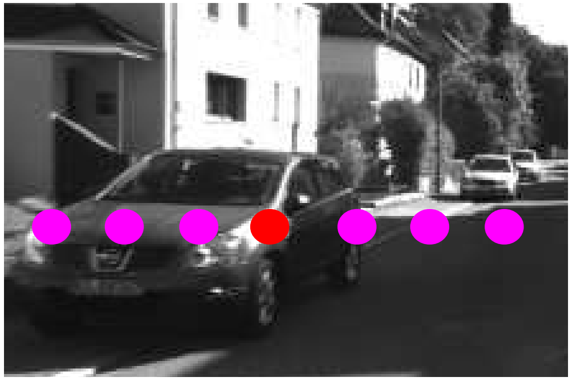


Option 2: Encoding = Gradients, Distance = Absolute Value

A little better. But still susceptible to scaling.

STEREO

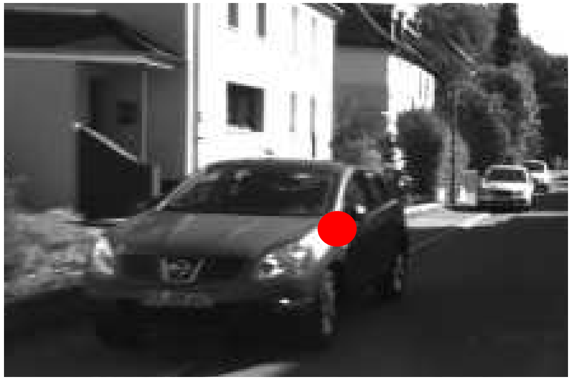


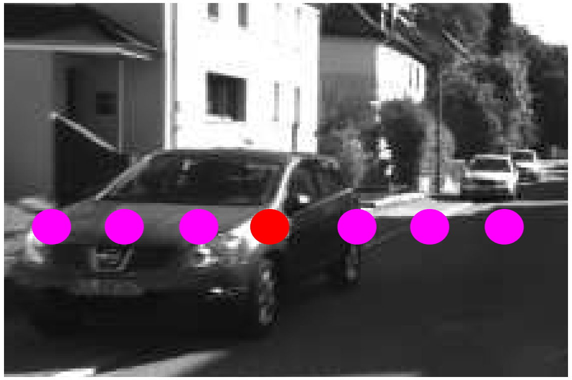


Option 3: Encoding = Clipped Gradients, Distance = Absolute Value
(between -X,X)

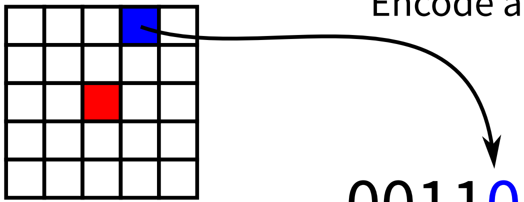
Better. Essentially becomes a test between "signs of gradients".

STEREO





Census Transform

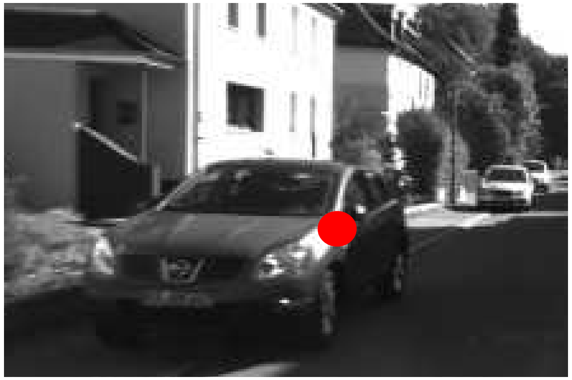


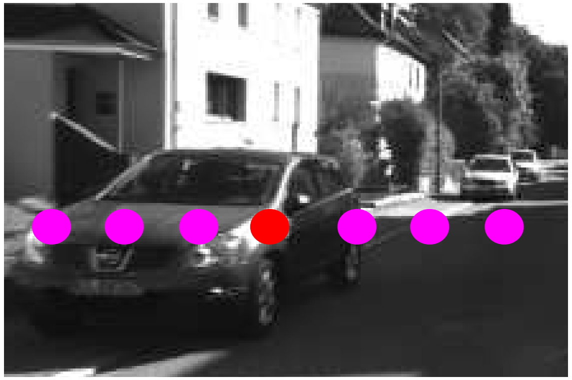
Encode a neighborhood as an integer, where each bit encodes if pixel at a different location had intensity higher than the center pixel.

0011000100001101

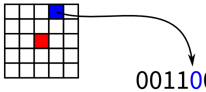
$$C[n] = \sum_j 2^j \delta\left(I[n] > I[n - n'_j]\right)$$

STEREO





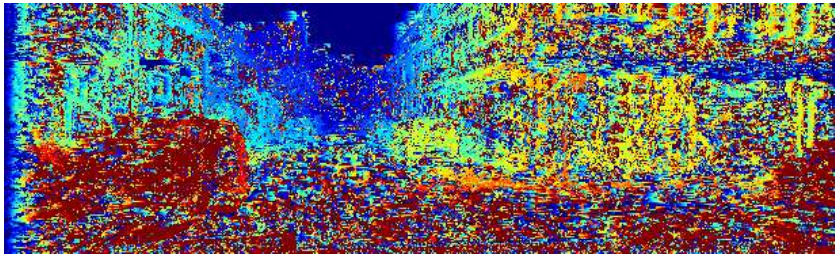
Step 1: At each location in left and right image, compute some representation of the appearance of (neighborhood around) that location.



Step 2: Define a "distance" function. Hamming Distance

Step 3: Determine disparity as the "best match" according to this distance.

STEREO



Still noisy: Some regions are inherently ambiguous, occlusions, ...

Use smoothness, left-right consistency, etc.